Internet Engineering Task Force
INTERNET DRAFT
DHC Working Group
Obsoletes: draft-ietf-dhc-dhcpv6-15.txt

J. Bound
Compaq Computer Corp.
M. Carney
Sun Microsystems, Inc
C. Perkins
Nokia Research Center
R. Droms(ed.)
Cisco Systems

22 November 2000

Dynamic Host Configuration Protocol for IPv6 (DHCPv6)
draft-ietf-dhc-dhcpv6-16.txt

Status of This Memo

Abstract

The Dynamic Host Configuration Protocol for IPv6 (DHCP) enables DHCP servers to pass configuration parameters such as IPv6 network addresses to IPv6 nodes. It offers the capability of automatic allocation of reusable network addresses and additional configuration flexibility. This protocol is a stateful counterpart to ``IPv6 Stateless Address Autoconfiguration'' [14], and can be used

separately or concurrently with the latter to obtain configuration parameters.

Contents

1. Introduction

This document describes DHCP for IPv6 (DHCP), a UDP [13] client / server protocol designed to reduce the cost of management of IPv6 nodes in environments where network managers require more control over the allocation of IPv6 addresses and configuration of network stack parameters than that offered by ``IPv6 Stateless Autoconfiguration'' [14]. DHCP is a stateful counterpart to stateless autoconfiguration. Note that both stateful and stateless autoconfiguration can be used concurrently in the same environment, leveraging the strengths of both mechanisms in order to reduce the cost of ownership and management of network nodes.

DHCP reduces the cost of ownership by centralizing the management of network resources such as IP addresses, routing information, OS installation information, directory service information, and other such information on a few DHCP servers, rather than distributing such information in local configuration files among each network node. DHCP is designed to be easily extended to carry new configuration parameters through the addition of new DHCP ``options'' defined to carry this information. (What were called ``extensions'' in the -15 draft are now called ``options''; see section 23.11.)

Those readers familiar with DHCP for IPv4 [6] will find DHCP for IPv6 provides a superset of features, and benefits from the additional features of IPv6 and freedom from BOOTP [4]-backward compatibility constraints. For more information about the differences between DHCP for IPv6 and DHCP for IPv4, see Appendix A.

This document is organized as follows. Section 2 defines terminology used throughout this document. Section 3 defines constant values used by DHCP. Section 4 briefly discusses requirement levels. Section 5 points the reader to helpful background specifications covering related IPv6 protocols. Section 6 discusses the design goals that influenced DHCP. Section 7 identifies some of the non-goals of this specification. Section 8 gives a high level overview of DHCP, its message types, and identifies DHCP functional entities (client, relay, server). Section 9 describes in detail the format of each DHCP message type. Section 10 discusses DHCP server solicitation. Section 11 discusses DHCP client-initiated configuration information exchange. Section 12 discusses DHCP server-initiated configuration information exchange. Section 14 presents helpful notes for DHCP client implementors. Section 15 presents helpful notes for DHCP server implementors. Section 16 presents helpful notes for DHCP relay implementors. Section 18 discusses security considerations for DHCP.

Section 23 describes the changes between this version of the DHCPv6 specification and draft-ietf-dhc-dhcpv6-15.txt.

2. Terminology

2.1. IPv6 Terminology

IPv6 terminology relevant to this specification from the IPv6 Protocol [5], IPv6 Addressing Architecture [7], and IPv6 Stateless Address Autoconfiguration [14] is included below.

address
: An IP layer identifier for an interface or a set of interfaces.

unicast address
: An identifier for a single interface. A packet sent to a unicast address is delivered to the interface identified by that address.

multicast address
: An identifier for a set of interfaces (typically belonging to different nodes). A packet sent to a multicast address is delivered to all interfaces identified by that address.

host
: Any node that is not a router.

IP
: Internet Protocol Version 6 (IPv6). The terms IPv4 and IPv6 are used only in contexts where it is necessary to avoid ambiguity.

interface
: A node's attachment to a link.

link
: A communication facility or medium over which nodes can communicate at the link layer, i.e., the layer immediately below IP. Examples are Ethernet (simple or bridged); Token Ring; PPP links, X.25, Frame Relay, or ATM networks; and Internet (or higher) layer "tunnels", such as tunnels over IPv4 or IPv6 itself.

link-layer identifier
: a link-layer identifier for an interface. Examples include IEEE 802 addresses for Ethernet or Token Ring network interfaces, and E.164 addresses for ISDN links.

link-local address
: An IP address having link-only scope, indicated by having the prefix (FE80::0000/64), that can be used to reach neighboring nodes attached to the same link. Every interface has a link-local address.

message A unit of data carried in a packet, exchanged between DHCP agents and clients.

neighbor A node attached to the same link.

node A device that implements IP.

packet An IP header plus payload.

prefix A bit string that consists of some number of initial bits of an address.

router A node that forwards IP packets not explicitly addressed to itself.

2.2. DHCP Terminology

Terminology specific to DHCP can be found below.

abort status
A status value returned to the application that has invoked a DHCP client operation, indicating anything other than success.

agent address
The address of a neighboring DHCP Agent on the same link as the DHCP client.

binding A binding (or, client binding) is a group of server data records indexed by <prefix, UUID> containing the server's information about the addresses and other information assigned to the IA.

DHCP Dynamic Host Configuration Protocol for IPv6. The terms DHCPv4 and DHCPv6 are used only in contexts where it is necessary to avoid ambiguity.

configuration parameter

An element of the configuration information set on the server and delivered to the client using DHCP. Such parameters may be used to carry information to be used by a node to configure its network subsystem and enable communication on a link or internetwork, for example.

DHCP client (or client)
A node that initiates requests on a link to obtain configuration parameters from one or more DHCP servers.

DHCP domain
A chunk of network topology managed by DHCP and operated by a single administrative entity.

DHCP server (or server)
A server is a node that responds to requests from clients, and may or may not be on the same link as the client(s).

DHCP relay (or relay)
A node that acts as an intermediary to deliver DHCP messages between clients and servers, and is on the same link as a client.

DHCP agent (or agent)
Either a DHCP server on the same link as a client, or a DHCP relay.

Identity association (IA)
A collection of addresses assigned to a client. Each IA has an associated UUID. A server identifies an IA by the tuple (prefix, UUID), where ``prefix'' is a prefix assigned to the link to which the client is attached, An IA may have 0 or more addresses associated with it.

Releasable resource
(Removed; see section 23.3.)

transaction-ID
An unsigned integer to match responses with replies initiated either by a client or server.

UUID
A universally unique identifier for a client.

DISCUSSION:

Rules for choosing a UUID are TBD.

3. DHCP Constants

This section describes various program and networking constants used by DHCP.

3.1. Multicast Addresses

DHCP makes use of the following multicast addresses:

All DHCP Agents address: FF02::1:2
This link-local multicast address is used by clients to communicate with the on-link agent(s) when they do not know those agents' link-local address(es). All agents (servers and relays) are members of this multicast group.

All DHCP Servers address: FF05::1:3
This site-local multicast address is used by clients or relays to communicate with server(s), either because they want to send messages to all servers or because they do not know the server(s) unicast address(es). Note that in order for a client to use this address, it must have an address of sufficient scope to be reachable by the server(s). All servers within the site are members of this multicast group.

3.2. UDP ports

DHCP uses the following destination UDP [13] port numbers. While source ports MAY be arbitrary, client implementations SHOULD permit their specification through a local configuration parameter to facilitate the use of DHCP through firewalls.

546 Client port. Used by agents to send messages to clients. Also used by servers to send messages to relays.

547 Agent port. Used by clients to send messages to agents. Also used by relays to send messages to servers.

3.3. DHCP message types

DHCP defines the following message types. More detail on these message types can be found in Section 9. Message types 0 and 9--255 are reserved and MUST be silently ignored.

01 DHCP Solicit

The DHCP Solicit (or Solicit) message is used by clients to locate servers. This message is multicast using the

All-DHCP-Agents address. Relay(s) forward Solicits as necessary to off-link servers.

Section 9.1 contains more details about the Solicit message.

02 DHCP Advertise

The DHCP Advertise (or Advertise) message is used by servers responding to Solicits. This message is unicast to the client's link-local address (if the server and client are on the same link) or unicast to the relay through which the Solicit was sent for final delivery to the client.

Section 9.2 contains more details about the Advertise message.

03 DHCP Request

The DHCP Request (or Request) message is used by clients to request configuration parameters from servers. This message is multicast using the All-DHCP-Agents address. Relay(s) forward Requests as necessary to off-link servers.

Section 9.3 contains more details about the Request message.

04 DHCP Reply

The DHCP Reply (or Reply) message is used by servers responding to Request and Release messages. In the case of responding to a Request message, the Reply contains configuration parameters destined for the client. This message is unicast to the client if the client has an address of sufficient scope that is reachable by the server. Otherwise, it is unicast to the relay through which the Request or Release message was sent for final delivery to the client.

Section 9.4 contains more details about the Reply message.

05 DHCP Release

The DHCP Release (or Release) message is used by clients to return one or more IP addresses to servers. The server will acknowledge the receipt of the Release message by sending the client a Reply message.

Section 9.5 contains more details about the Release message.

06 DHCP Reconfigure

07 DHCP Reconfigure-reply

Removed; see section 23.2.

08 DHCP Reconfigure-init

The DHCP Reconfigure-init (or Reconfigure-init) message is set by server(s) to inform client(s) that the server(s) has new or updated configuration parameters, and that the client(s) are to initiate a Request/Reply transaction with the server(s) in order to receive the updated information.

Section 9.8 contains more details about the Reconfigure-init message.

3.4. Error Values

This section describes error values exchanged between DHCP implementations.

3.4.1. Generic Error Values

The following symbolic names are used between client and server implementations to convey error conditions. The following table contains the actual numeric values for each name. Note that the numeric values do not start at 1, nor are they consecutive. The errors are organized in logical groups.

Error Name	Error ID	Description
Success	00	Success
UnspecFail	16	Failure, reason unspecified
AuthFailed	17	Authentication failed or nonexistent
PoorlyFormed	18	Poorly formed message
Unavail	19	Addresses unavailable

3.4.2. Server-specific Error Values

The following symbolic names are used by server implementations to convey error conditions to clients. The following table contains the actual numeric values for each name.

Error_Name	Error_ID	Description
NoBinding	20	Client record (binding) unavailable
InvalidSource	21	Invalid Client IP address
NoServer	23	Relay cannot find Server Address
ICMPError	64	Server unreachable (ICMP error)

3.5. Configuration Variables

This section presents a table of client and server configuration variables and the default or initial values for these variables. The client-specific variables MAY be configured on the server and MAY be delivered to the client through the ``DHCP Retransmission Parameter Option'' in a Reply message. This option is TBD.

Parameter	Default	Description
MIN_SOL_DELAY	1	MIN (secs) to delay 1st mesg
MAX_SOL_DELAY	5	MAX (secs) to delay 1st mesg
ADV_MSG_TIMEOUT	500	SOL Retrans timer (msecs)
ADV_MSG_MAX	30	MAX timer value (secs)
SOL_MAX_ATTEMPTS	-1	MAX attempts (-1 = infinite)
REP_MSG_TIMEOUT	250	REQ Retrans timer (msecs)
REQ_MSG_ATTEMPTS	10	MAX Request attempts
REL_MSG_ATTEMPTS	5	MAX Release attempts
RECREP_MSG_TIMEOUT	2000	Retrans timer (msecs)
REC_MSG_ATTEMPTS	10	Reconfigure attempts
REC_REP_MIN	5	Minimum pause interval (secs)
REC_REP_MAX	7200	Maximum pause interval (secs)
REC_THRESHOLD	100	% of required clients
SRVR_PREF_WAIT	2	Advertise Collect timer (secs)

4. Requirements

The keywords MUST, MUST NOT, REQUIRED, SHALL, SHALL NOT, SHOULD, SHOULD NOT, RECOMMENDED, MAY, and OPTIONAL, when they appear in this document, are to be interpreted as described in [2].

This document also makes use of internal conceptual variables to describe protocol behavior and external variables that an implementation must allow system administrators to change. The specific variable names, how their values change, and how their settings influence protocol behavior are provided to demonstrate protocol behavior. An implementation is not required to have them in the exact form described here, so long as its external behavior is consistent with that described in this document.

5. Background

Related work in IPv6 that would best serve an implementor to study is the IPv6 Specification [5], the IPv6 Addressing Architecture [7], IPv6 Stateless Address Autoconfiguration [14], IPv6 Neighbor Discovery Processing [11], and Dynamic Updates to DNS [16]. These specifications enable DHCP to build upon the IPv6 work to provide both robust stateful autoconfiguration and autoregistration of DNS Host Names.

The IPv6 Specification provides the base architecture and design of IPv6. A key point for DHCP implementors to understand is that IPv6 requires that every link in the Internet have an MTU of 1280 octets or greater (in IPv4 the requirement is 68 octets). This means that a UDP packet of 536 octets will always pass through an internetwork (less 40 octets for the IPv6 header), as long as there are no IP options prior to the UDP header in the packet. But, IPv6 does not support fragmentation at routers, so that fragmentation takes place end-to-end between hosts. If a DHCP implementation needs to send a packet greater than 1500 octets it can either fragment the UDP packet into fragments of 1500 octets or less, or use Path MTU Discovery [9] to determine the size of the packet that will traverse a network path.

DHCP clients use Path MTU discovery when they have an address of sufficient scope to reach the DHCP server. If a DHCP client does not have such an address, that client MUST fragment its packets if the resultant message size is greater than the minimum 1280 octets.

Path MTU Discovery for IPv6 is supported for both UDP and TCP and can cause end-to-end fragmentation when the PMTU changes for a destination.

The IPv6 Addressing Architecture specification [7] defines the address scope that can be used in an IPv6 implementation, and the various configuration architecture guidelines for network designers of the IPv6 address space. Two advantages of IPv6 are that support for multicast is required, and nodes can create link-local addresses during initialization. This means that a client can immediately use its link-local address and a well-known multicast address to begin communications to discover neighbors on the link. For instance, a client can send a Solicit message and locate a server or relay.

IPv6 Stateless Address Autoconfiguration [14] (Addrconf) specifies procedures by which a node may autoconfigure addresses based on router advertisements [11], and the use of a valid lifetime to support renumbering of addresses on the Internet. In addition the protocol interaction by which a node begins stateless or stateful autoconfiguration is specified. DHCP is one vehicle to perform stateful autoconfiguration. Compatibility with addrconf is a design requirement of DHCP (see Section 6).

IPv6 Neighbor Discovery [11] is the node discovery protocol in IPv6 which replaces and enhances functions of ARP [12]. To understand IPv6 and Addrconf it is strongly recommended that implementors understand IPv6 Neighbor Discovery.

Dynamic Updates to DNS [16] is a specification that supports the dynamic update of DNS records for both IPv4 and IPv6. DHCP can use the dynamic updates to DNS to integrate addresses and name space to not only support autoconfiguration, but also autoregistration in IPv6. The security model to be used with DHCPv6 should conform as closely as possible to the authentication model outlined in RFC2402 [8].

6. Design Goals

- DHCP is a mechanism rather than a policy. Network administrators set their administrative policies through the configuration parameters they place upon the DHCP servers in the DHCP domain they're managing. DHCP is simply used to deliver parameters according to that policy to each of the DHCP clients within the domain.

- DHCP is compatible with IPv6 stateless autoconf [14].

- DHCP does not require manual configuration of network parameters on DHCP clients, except in cases where such configuration is needed for security reasons. A node configuring itself using DHCP should require no user intervention.

- DHCP does not require a server on each link. To allow for scale and economy, DHCP must work across DHCP relays.

- DHCP coexists with statically configured, non-participating nodes and with existing network protocol implementations.

- DHCP clients can operate on a link without IPv6 routers present.

- DHCP will provide the ability to renumber network(s) when required by network administrators [3].

- A DHCP client can make multiple, different requests for configuration parameters when necessary from one or more DHCP servers at any time.

- DHCP will contain the appropriate time out and retransmission mechanisms to efficiently operate in environments with high latency and low bandwidth characteristics.

7. Non-Goals

This specification explicitly does not cover the following:

- Specification of a DHCP server to server protocol.

- How a DHCP server stores its DHCP data.

- How to manage a DHCP domain or DHCP server.

- How a DHCP relay is configured or what sort of information it may log.

8. Overview

This section provides a general overview of the interaction between the functional entities of DHCP. The overview is organized as a series of questions and answers. Details of DHCP such as message formats and retransmissions are left to sections 9, 10, 11, 12, 14, 15, and 16.

8.1. How does a node know to use DHCP?

An unconfigured node determines that it is to use DHCP for configuration of an interface by detecting the presence (or absence) of routers on the link. If router(s) are present, the node examines router advertisements to determine if DHCP should be used to configure the interface. If there are no routers present, then the node MUST use DHCP to configure the interface. Detail on this process can be found in neighbor discovery [11] and stateless autoconfiguration [14].

8.2. How does a client find out about DHCP agents?

(Section removed, see 23.6

8.3. What if the client and server(s) are on different links?

Use of DHCP in such environments requires one or more DHCP relays be set up on the client's link, because a client may only have a

link-local address. Relays receive the Solicit and Request messages from the client and forward them to some set of servers within the DHCP domain. The client message is forwarded verbatim as the payload in a message from the relay to the server. A relay will include one of its own addresses (of sufficient scope) from the interface on the same link as the client, as well as the prefix length of that address, in its message to the server. Servers receiving the forwarded traffic use this information to aid in selecting configuration parameters appropriate to the client's link. The servers also use the relay's address as the destination to forward client-destined messages for final delivery by the relay.

Relays forward client messages to servers using some combination of the FF05::1:3(All Servers) site-local multicast address, some other (perhaps a combination) of site-local multicast addresses set up within the DHCP domain to include the servers in that domain, or a list of unicast addresses for servers. The network administrator makes relay configuration decisions based upon the topological requirements (scope) of the DHCP domain they are managing. Note that if the DHCP domain spans more than the site-local scope, then the relays MUST be configured with global addresses for the client's link so as to be reachable by servers outside the relays' site-local environment.

8.4. How does a client request configuration parameters from servers?

To request configuration parameters, the client forms a Request message, and sends it to the server either directly (client has an address of sufficient scope) or indirectly (through the on-link relay). The client MAY include a Option Request Option 22.3 (ORO) along with other options to request specific information from the server. Note that the client MAY form multiple Request messages and send each of them to different servers to request potentially different information (perhaps based upon what was advertised) in order to satisfy its needs. As a client's needs may change over time (perhaps based upon an application's requirements), the client may form additional Request messages to request additional information as it is needed.

The server(s) respond with Reply messages containing the requested configuration parameters, which can include status information regarding the information requested by the client. The Reply MAY also include additional information, such as a reconfiguration event multicast group for the client to join to monitor reconfiguration events, as described in section 8.8.

8.5. How do clients and servers identify and manage addresses?

Servers and clients manage addresses in groups called ``identity associations.'' Each identity associations is identified using a unique identifier. An identity association may contain one or more IPv6 addresses. DHCP servers assign addresses to identity associations. DHCP clients use the addresses in an identity association to configure interfaces. There is always at least one identity association per interface that a client wishes to configure. Each address in an IA has its own preferred and valid lifetime. Over time, the server may change the characteristics of the addresses in an IA; for example, by changing the preferred or valid lifetime for an address in the IA. The server may also add or delete addresses from an IA; for example, deleting old addresses and adding new addresses to renumber a client. A client can request the current list of addresses assigned to an IA from a server through an exchange of protocol messages.

8.6. Can a client release its assigned addresses before the lease expires?

A client forms a Release message, including options identifying the IA to be released. The client sends the Release to the server which assigned the addresses to the client initially. If that server cannot be reached after a certain number of attempts (see section 3.5), the client can abandon the Release attempt. In this case, the address(es) in the IA will be reclaimed by the server(s) when the lifetimes on the addresses expire.

8.7. What if the client determines one or more of its assigned addresses are already being used by another client?

If the client determines through a mechanism like Duplicate Address Detection [14] that the address it was assigned by the server is already in use by another client, the client will form a Release message, including the option carrying the in-use address. The option's status field MUST be set to the value reflecting the ``in use'' status of the address.

8.8. How are clients notified of server configuration changes?

There are two possibilities. Either the clients discover the new information when they revisit the server(s) to request additional configuration information / extend the lifetime on an address. or through a server-initiated event known as a reconfigure event.

The reconfiguration feature of DHCP offers network administrators the opportunity to update configuration information on DHCP clients whenever necessary. To signal the need for client reconfiguration, the server will unicast a Reconfigure-init message to each client individually. The server may use multicast to signal the reconfiguration to multiple clients simultaneously. (Note that there is no mechanism defined in the protocol to guarantee that every client actually performs a reconfiguration in response to a multicast reconfigure-init message.) A Reconfigure-init is a trigger which will cause the client(s) to initiate a standard Request/Reply exchange with the server in order to acquire the new or updated addresses.

9. Message Formats and Identity Associations

All reserved fields in a message MUST be transmitted as zeroes and ignored by the receiver of the message.

DISCUSSION:

Each DHCP message has an identical fixed format header; some messages also allow a variable format area for options. Not all fields in the header are used in every message. In this section, every field is included in every message format diagram and fields that are not used in a message are marked as ``unused''. As an alternative, the unused fields could be labeled ``unused'' in the format diagram.

9.1. DHCP Solicit Message Format

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  msg-type = 1 |  preference   |         transaction-ID        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                   client-link-local-address                   |
|                          (16 octets)                          |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                         server-address                        |
|                          (16 octets)                          |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

preference
(unused) MUST be 0

transaction-ID
An unsigned integer generated by the client used to identify this Solicit message.

client-link-local-address
The link-local address of the interface for which the client is using DHCP.

server-address (unused) MUST be 0

9.2. DHCP Advertise Message Format

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  msg-type = 2 |  preference   |         transaction-ID        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                   client-link-local-address                   |
|                          (16 octets)                          |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                         server-address                        |
|                          (16 octets)                          |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|             options (variable number and length)   ....       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

preference An unsigned integer indicating a server's willingness to provide service to the client.

transaction-ID An unsigned integer used to identify this Advertise message. Copied from the client's Solicit message.

client-link-local-address
The IP link-local address of the client interface from which the client issued the Solicit message.

server-address
The IP address of the server. If the DHCP domain

crosses site boundaries, then this address MUST be globally-scoped.

options Options are described elsewhere in this document

See Sections 14.4 and 15.3 for information about how clients and servers handle the preference field.

9.3. DHCP Request Message Format

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  msg-type = 3 |  preference   |         transaction-ID        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                   client-link-local-address                   |
|                          (16 octets)                          |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                         server-address                        |
|                          (16 octets)                          |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|            options (variable number and length)   ....        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

preference
(unused) MUST be 0

transaction-ID
An unsigned integer generated by the client used to identify this Request message.

client-link-local-address
The link-local address of the client interface from which the client will issue the Request message.

server-address
The IP address of the server to which the the client's Request message is directed, copied from an Advertise message.

options
Options are described elsewhere in this document.

9.4. DHCP Reply Message Format

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |  msg-type = 4 |  preference   |         transaction-ID        |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                                                               |
   |                      client-link-local-address                |
   |                            (16 octets)                        |
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                                                               |
   |                          server-address                       |
   |                           (16 octets)                         |
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |            options (variable number and length)   ....        |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

preference An unsigned integer indicating a server's willingness to provide service to the client.

transaction-ID
An unsigned integer used to identify this Reply message. Copied from the client's Request message.

client-link-local-address
The link-local address of the interface for which the client is using DHCP.

server-address
The IP address of the server. If the DHCP domain crosses site boundaries, then this address MUST be globally-scoped.

options
Options are described elsewhere in this document.

9.5. DHCP Release Message Format

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  msg-type = 5 | preference    |        transaction-ID         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                   client-link-local-address                   |
|                          (16 octets)                          |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                         server-address                        |
|                          (16 octets)                          |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         options (variable number and length)   ....         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

preference (unused) MUST be 0

transaction-ID
An unsigned integer generated by the client used to identify this Release message.

P (unused) MUST be 0

client-link-local-address
The client's link-local address for the interface from which the client issued the Release message.

server-address
The IP address of the server that assigned the addresses.

options See section 22.

9.6. DHCP Reconfigure Message Format

The Reconfigure message has been deleted (see section 23.2).

9.7. DHCP Reconfigure-reply Message Format

The Reconfigure-reply message has been deleted (see section 23.2).

9.8. DHCP Reconfigure-init Message Format

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  msg-type = 8 |  preference   |         transaction-ID        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                   client-link-local-address                   |
|                          (16 octets)                          |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                        server-address                         |
|                          (16 octets)                          |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|            options (variable number and length)   ....        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

preference (unused) MUST be 0

transaction-ID
: An unsigned integer generated by the server to identify this Reconfigure-init message

client-link-local-address
: (unused) MUST be 0

server-address
: The IP address of the DHCP server issuing the Reconfigure-init message. MUST be of sufficient scope to be reachable by all clients.

options
: SHOULD only include an ``Options request option'' (ORO) and/or authentication options. No configuration information SHOULD be included. See section 22 more information about options.

9.9. Relay-forward message

```
      0                   1                   2                   3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |  msg-type TBD | prefix length |                               |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+                               |
     |                                                               |
     |                         relay-address                         |
     |                                                               |
     |                               |-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-|
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+                               |
     |            options (variable number and length)   ....        |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

      msg-type   TBD

      prefix-length
                 The length of the prefix in the address in the
                 ``relay-address'' field.

      relay-address
                 An address assigned to the interface through which the
                 message from the client was received.

      options   MUST include a ``Client message option''; see
                 section 22.4.

9.10. Server-forward message

```
      0                   1                   2                   3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |  msg-type TBD | prefix length |                               |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+                               |
     |                                                               |
     |                         relay-address                         |
     |                                                               |
     |                               |-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-|
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+                               |
     |            options (variable number and length)   ....        |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

      msg-type   TBD

prefix-length
The length of the prefix in the address in the ``relay-address'' field.

relay-address
An address identifying the interface through which the message from the server should be forwarded; copied from the ``client-forward'' message.

options MUST include a ``Server message option''; see section 22.5.

9.11. Identity association

An ``identity-association'' (IA) is a construct through which a server and a client can identify, group and manage IPv6 addresses. Each IA consists of a UUID and a list of associated IPv6 addresses (the list may be empty). A client associates an IA with one of its interfaces and uses the IA to obtain IPv6 addresses for that interface from a server.

10. DHCP Server Solicitation

This section describes how a client locates servers. The behavior of client, server, and relay implementations is discussed, along with the messages they use.

(Prefix advertisements have been deleted; see 23.9.)

10.1. Solicit Message Validation

Clients MUST silently discard any received Solicit messages.

Agents MUST silently discard any received Solicit messages if the ``client-link-local-address'' field does not contain a valid link-local address.

10.2. Advertise Message Validation

Servers MUST discard any received Advertise messages.

Clients MUST discard any Advertise messages that meet any of the following criteria:

- o The ``Transaction-ID'' field value does not match the value the client used in its Solicit message.

- o The ``client-link-local-address'' field value does not match the link-local address of the interface upon which the client sent the Solicit message.

10.3. Client Behavior

Clients use the Solicit message to discover DHCP servers configured to serve addresses on the link to which the client is attached.

(Prefix advertisement by servers has been deleted; see section 23.9.)

10.3.1. Creation and sending of the Solicit message

The client sets the ``msg-type'' field to 1, and places the link-local address of the interface it wishes to configure in the ``client-link-local-address'' field. The client sets all other fields to zero.

The client sends the Solicit message to the FF02::1:2 (All DHCP Agents) multicast address, destination port 547. The source port selection can be arbitrary, although it SHOULD be possible using a client configuration facility to set a specific source port value.

10.3.2. Time out and retransmission of Solicit Messages

The client's first Solicit message on the interface MUST be delayed by a random amount of time between the interval of MIN_SOL_DELAY and MAX_SOL_DELAY. This random delay desynchronizes clients which start at the same time (e.g., after a power outage).

The client waits ADV_MSG_TIMEOUT, collecting Advertise messages. If no Advertise messages are received, the client retransmits the Solicit, and doubles the ADV_MSG_TIMEOUT value. This process continues until either one or more Advertise messages are received or ADV_MSG_TIMEOUT reaches the ADV_MSG_MAX value. Thereafter, Solicits are retransmitted every ADV_MSG_MAX until SOL_MAX_ATTEMPTS have been made, at which time the client stops trying to DHCP configure the interface. An event external to DHCP is required to restart the DHCP configuration process.

Default and initial values for MIN_SOL_DELAY, MAX_SOL_DELAY, ADV_MSG_TIMEOUT, AND ADV_MSG_MAX are documented in section 3.5.

10.3.3. Receipt of Advertise messages

Upon receipt of one or more validated Advertise messages, the client selects one or more Advertise messages based upon the following criteria.

- Those Advertise messages with the highest server preference value (see section 14.4) are preferred over all other Advertise messages.

- Within a group of Advertise messages with the same server preference value, a client MAY select those servers whose Advertise messages advertise information of interest to the client. For example, one server may be advertising the availability of IP addresses which have an address scope of interest to the client.

Once a client has selected Advertise message(s), the client will typically store information about each server, such as server preference value, addresses advertised, when the advertisement was received, and so on. Depending on the requirements of the client's invoking user, the client MAY initiate a configuration exchange with the server(s) immediately, or MAY defer this exchange until later.

10.4. Relay Behavior

For this discussion, the Relay may be configured to use a list of server destination addresses, which may include unicast addresses, the FF05::1:3 (All DHCP Servers) multicast address, or other multicast addresses selected by the network administrator. If the Relay has not been explicitly configured, it will use the FF05::1:3 (All DHCP Servers) multicast address as the default.

10.4.1. Relaying of Solicit messages

When a Relay receives a valid Solicit message, it constructs a Relay-forward message. The client Solicit message is carried as the payload of a ``client-message'' option. The relay places an address from the interface on which the Solicit message was received in the ``relay-address'' field and the prefix length for that address in the ``prefix-length'' field. The Relay then sends the Relay-forward message to the list of server destination addresses that it has been configured with.

10.4.2. Relaying of Advertise messages

When the relay receives a Relay-reply message, it extracts the server message from the ``server-message'' option and forwards the server message to the address in the client-link-local-address field in the server message. The relay forwards the server message through the interface identified in the ``relay-address'' field in the Relay-reply message.

10.5. Server Behavior

For this discussion, the Server is assumed to have been configured in an implementation specific manner. This configuration is assumed to contain all network topology information for the DHCP domain, as well as any necessary authentication information.

10.5.1. Receipt of Solicit messages

If the server receives a Solicit message, the client must be on the same link as the server. If the server receives a Relay-forward message containing a Solicit message, the client must be on the link to which the prefix identified by the ``relay-address'' and ``prefix-length'' fields in the Relay-forward message is assigned. The server records the ``relay-address'' field from the Relay-forward message and extracts the solicit message from the ``client-message'' option.

If administrative policy permits the server to respond to a client on that link, the server will generate and send an Advertise message to the client.

10.5.2. Creation and sending of Advertise messages

The server sets the ``msg-type'' field to 2 and copies the values of the following fields from the client's Solicit to the Advertise message:

- transaction-ID
- client-link-local-address

The server places one of its IP addresses (determined through administrator setting) in the ``server-address'' field of the Advertise message. The server sets the ``preference'' field according to its configuration information. See section 15.3 for a description of server preference.

If the Solicit message was received in a Relay-forward message, the server constructs a Relay-reply message with the Advertise message in the payload of a ``server-message'' option. The server unicasts the Relay-reply message to the address in the ``relay-address'' field from the Relay-forward message.

If the Solicit message was received directly by the server, the server unicasts the Advertise message directly to the client using the ``client-link-local-address'' field value as the destination address. The Advertise message MUST be unicast through the interface on which the Solicit message was received.

DISCUSSION:

(From Ted Lemon) There is a danger in using Solicit versus DHCPDISCOVER: in the Solicit paradigm, the client has to choose the DHCP server before it knows if the DHCP server will give it an IP address, or which addresses the server is willing to assign to the client. It may be that there are two or more DHCP servers owned by the same administrative domain, and both are theoretically willing to give the client addresses, but only one actually has any addresses to give.

11. DHCP Client-Initiated Configuration Exchange

A client uses the Request-Reply message exchange to acquire configuration information of interest. The client may initiate the configuration exchange as part of the operating system configuration process or when requested to do so by the application layer.

A client uses the Release-Reply message exchange to indicate to the DHCP server that the client will no longer be using the addresses in the released IA.

11.1. Request Message Validation

Clients MUST silently discard any received Request messages.

Agents MUST discard any Request messages in which the ``client-link-local-address'' field does not contain a valid link-local address.

Servers MUST discard any received Request message which meets any of the following criteria:

- o The ``server-address'' field value does not match any of the server's addresses.

- o The ``options'' field contains an authentication option, and the server cannot successfully authenticate the client.

11.2. Reply Message Validation

Servers MUST silently discard any received Reply messages.

Clients MUST discard any Reply message that meets any of the following criteria:

- o The ``transaction-ID'' field value does not match the value the client used in its Request or Release message.

- o The ``client-link-local-address'' field value does not match the link-local address of the interface upon which the client sent in its Request or Release message.

- o The Reply message contains an authentication option, and the client's attempt to authenticate the message fails.

Relays MUST discard any Relay-reply message in which the ``client-link-local-address'' in the encapsulated Reply message does not contain a valid link-local address.

11.3. Release Message Validation

Clients MUST silently discard any received Release messages.

Agents MUST discard any Release message in which the ``client-link-local-address'' field does not contain a valid link-local address.

Servers MUST discard any received Release message in which the ``options'' field contains an authentication option, and the server cannot successfully authenticate the client.

11.4. Client Behavior

A client will generate one or more Request messages to acquire configuration information. A client may initiate such an exchange automatically in order to acquire the necessary network parameters to communicate with nodes off-link. The client uses the server address information from previous Advertise message(s) for use in

constructing Request message(s). Note that a client may request configuration information from one or more servers at any time.

A client uses the Release message in the management of IAs when:

- o The client has determined through DAD or some other method that one or more of the addresses assigned by the server in the IA is already in use by a different client.

- o The client has been instructed to release the IA prior to the IA expiration time since it is no longer needed.

11.4.1. Creation and sending of Request messages

The client sets the ``msg-type'' field to 3, and places the link-local address of the interface it wishes to acquire configuration information for in the ``client-link-local-address'' field.

The client generates a transaction ID inserts this value in the ``transaction-ID'' field.

The client places the address of the destination server in the ``server-address'' field.

The client adds any appropriate options, including one or more IA options (if the client is requesting that the server assign it some network addresses). If the client does include any IA options, it MUST include the list of addresses the client currently has associated with that IA. If the client is requesting configuration of a new IA, the list of addresses MUST be empty.

11.4.2. Time out and retransmission of Request Messages

The server will respond to the Request message with a Reply message. If no Reply message is received within REP_MSG_TIMEOUT milliseconds, the client retransmits the Request with the same transaction-ID, and doubles the REP_MSG_TIMEOUT value, and waits again. The client continues this process until a Reply is received or REQUEST_MSG_ATTEMPTS unsuccessful attempts have been made, at which time the client MUST abort the configuration attempt. The client SHOULD report the abort status to the application layer.

Default and initial values for REP_MSG_TIMEOUT and REQ_MSG_ATTEMPTS are documented in section 3.5.

11.4.3. Receipt of Reply message in response to a Request

Upon the receipt of a valid Reply message, the client extracts the configuration information contained in the Reply. If the ``status'' field contains a non-zero value, the client reports the error status to the application layer.

The client records the T1 and T2 times for each IA in the Reply message. The client records any addresses included with IAs in the Reply message. The client updates the preferred and valid lifetimes for the addresses in the IA from the lifetime information in the IA option. The client leaves any addresses that the client has associated with the IA that are not included in the IA option unchanged.

Management of the specific configuration information is detailed in the definition of each option, in section 22.

11.4.4. Creation and sending of Release messages

The client sets the ``msg-type'' field to 5, and places the link-local address of the interface associated with the configuration information it wishes to release in the ``client-link-local-address'' field.

The client generates a transaction ID and places this value in the ``transaction-ID'' field.

The client includes options containing the IAs it is releasing in the ``options'' field. The appropriate ``status'' field in the options MUST be set to indicate the reason for the release.

The client places the IP address of the server that allocated the address(es) in the ``server-address'' field.

If the client is configured to use authentication, the client generates the appropriate authentication option, and adds this option to the ``options'' field. Note that the authentication option MUST be the last option in the ``options'' field. See section 22.7 for more details about the authentication option.

(The client always forwards Release messages to the server through a relay; see section 11.5.)

11.4.5. Time out and retransmission of Release Messages

If no Reply message is received within REP_MSG_TIMEOUT milliseconds, the client retransmits the Release, doubles the REP_MSG_TIMEOUT value, and waits again. The client continues this process until a Reply is received or REL_MSG_ATTEMPTS unsuccessful attempts have been made, at which time the client SHOULD abort the release attempt. The client SHOULD return the abort status to the application, if an application initiated the release.

Default and initial values for REP_MSG_TIMEOUT and REL_MSG_ATTEMPTS are documented in section 3.5.

Note that if the client fails to release the IA, the addresses assigned to the IA will be reclaimed by the server when the lease associated with it expires.

11.4.6. Receipt of Reply message in response to a Release

Upon receipt of a valid Reply message, the client can consider the Release event successful, and SHOULD return the successful status to the application layer, if an application initiated the release.

11.4.7. When a client should send a Request message

The description of the Request/Reply message exchange in this section makes no assumptions about the timing or state of the client when it initiates a Request/Reply message exchange. Sections 11.4.8 through 11.4.10 describe when a client MAY initiate a Request/Reply message exchange. The procedures for timeout and retransmission of Request messages are described in section 11.4.2.

11.4.8. Initialization

If a client has no valid IPv6 addresses of sufficient scope to communicate with a DHCP server, it may a Request message to obtain new addresses. The client includes one or more IAs in the Request message, to which the server assigns new addresses. The server then returns to IA(s) to the client in a Reply message.

11.4.9. Confirming the validity of IPv6 addresses

Whenever a client may have moved to a new link, its IPv6 addresses may no longer be valid. Examples of times when a client may have moved to a new link include:

- o The client reboots
- o The client is physically disconnected from a wired connection
- o The client returns from sleep mode
- o The client using a wireless technology changes cells

In any situation when a client may have moved to a new link, the client MUST initiate a Request/Reply message exchange. The client includes any IAs, along with the addresses associated with those IAs, in its Request message. The server returns the IAs with updated list of addresses and associated lifetimes.

11.4.10. Extending the lifetimes on IPv6 addresses

IPv6 addresses assigned to a client through an IA use the same preferred and valid lifetimes as IPv6 addresses obtained through stateless autoconfiguration. The server assigns preferred and valid lifetimes to the IPv6 addresses it assigns to an IA. To extend those lifetimes, the client sends a Request to the server containing an ``IA option'' for the IA and its associated addresses. The server determines new lifetimes for the addresses in the IA according to the server's administrative configuration. The server may also add new addresses to the IA. The server remove addresses from the IA by setting the preferred and valid lifetimes of those addresses to zero.

The server controls the time at which the client contacts the server to extend the lifetimes on assigned addresses through the T1 and T2 parameters assigned to an IA. If the server does not assign an explicit value to T1 or T2 for an IA, T1 defaults to 0.5 times the shortest preferred lifetime of any address assigned to the IA and T2 defaults to 0.875 times the shortest preferred lifetime of any address assigned to the IA.

At time T1 for an IA, the client initiates a Request/Reply message exchange to extend the lifetimes on any addresses in the IA. The client includes an IA option with all addresses currently assigned to the IA in its Request message. The client unicasts this Request message to the server that originally assigned the addresses to the IA.

At time T2 for an IA (which will only be reached if the server to which the Request message was sent at time T1 has not responded), the client initiates a Request/Reply message exchange. The client includes an IA option with all addresses currently assigned to the IA in its Request message. The client multicasts this message to the FF02::1:2 (All DHCP Agents) multicast address.

11.5. Relay Behavior

11.5.1. Relaying of Request or Release messages

When a Relay receives a valid Request or Release message, it constructs a Relay-forward message. The client message is carried as the payload of a ``client-message'' option. The relay places an address from the interface on which the client message was received in the ``relay-address'' field and the prefix length for that address in the ``prefix-length'' field. The Relay then forwards the Relay-forward message to the list of server destination addresses that it has been configured with.

11.6. Server Behavior

For this discussion, the Server is assumed to have been configured in an implementation specific manner with configuration of interest to clients.

11.6.1. Receipt of Request messages

Upon the receipt of a valid Request message from a client the server can respond to, (implementation-specific administrative policy satisfied) the server scans the options field.

The server then constructs a Reply message and sends it to the client.

DISCUSSION:

This section needs text about managing IAs and determining options to be returned to client.

11.6.2. Receipt of Release messages

Upon the receipt of a valid Release message, the server examines the IAs and the addresses in the IAs for validity. If the IAs in the message are in a binding for the client and the addresses in the IAs have been assigned by the server to those IA, the server deletes the addresses from the IAs and makes the addresses available for assignment to other clients.

The server then generates a Reply message. If all of the IAs were valid and the addresses successfully released,, the server sets the ``status'' field to ``Success''. If any of the IAs were invalid or if any of the addresses were not successfully released, the server

releases none of the addresses in the message and sets the ``status'' field to ``NoBinding''(section 3.4).

DISCUSSION:

What is the behavior of the server relative to a ``partially released'' IA; i.e., an IA for which some but not all addresses are released?

Can a client send an empty IA to release all addresses in the IA?

If the IA becomes empty - all addresses are released - can the server discard any record of the IA?

11.6.3. Creation and sending of Reply messages

DISCUSSION:

XXX - This section needs to be fixed (see section 11.6.1).

The server sets the ``msg-type'' field to 4 and copies the values of the following fields from the client's Request or Release to the Reply message:

- o transaction-ID
- o client's link-local address
- o server-address

The server sets the ``status'' field appropriately (see the table in section 3.4) based upon the results of processing the client's request.

If the Request or Release message from the client was originally received by the server, the server unicasts the Reply message to the link-local address in the ``client-link-local-address'' field.

If the message was originally received in a Forward-request or Forward-release message from a relay, the server places the Reply message in the options field of a Response-reply message and unicasts the message to the relay's address from the original message.

12. DHCP Server-Initiated Configuration Exchange

A server initiates a configuration exchange on behalf of the administrator of the DHCP domain. An administrator may initiate such an exchange when new links are added to the domain or existing links are to be renumbered. Other examples include changes in the location of directory servers, addition of new services such as printing, and availability of new software (system or application).

DISCUSSION:

Changed ``networks'' to ``links'' here (ed.). Why would adding new links cause a server-initiated configuration exchange?

12.1. Reconfigure Message Validation

Reconfigure messages have been deleted; see section 23.2.

12.2. Reconfigure-reply Message Validation

Reconfigure-reply messages have been deleted; see section 23.2.

12.3. Reconfigure-init Message Validation

Agents MUST silently discard any received Reconfigure-init messages.

Clients MUST discard any Reconfigure-init messages that do not contain an authentication option or that fail the client's authentication check.

12.4. Server Behavior

For this discussion, the server is assumed to have a implementation-specific interface by which an administrator may initiate a reconfiguration event with some set of clients.

A server sends a Reconfigure-init message to trigger a client to initiate immediately a Request/Reply message exchange with the server. A server can send Reconfigure-init messages only to those clients who have an address of sufficient scope to be reachable by the server. Thus, those clients who have not requested an IP address and are off-link cannot be reconfigured by the server.

DISCUSSION:

It would be possible to forward Reconfigure-init messages through relays if the server records the client's link-local address and the relay's address from the client's Request message.

12.4.1. Creation and sending of Reconfigure messages

Reconfigure messages have been deleted; see section 23.2.

12.4.2. Time out and retransmission of Reconfigure messages

12.4.3. Receipt of Reconfigure-reply messages

12.4.4. Creation and sending of Reconfigure-init messages

The server sets the ``msg-type'' field to 8. The server generates a transaction-ID and inserts it in the ``transaction-ID'' field. The server places its address (of appropriate scope) in the ``server-address'' field.

The server MAY include an ORO option to inform the client of what information has been changed or new information that has been added.

The server MUST include an authentication option with the appropriate settings and add that option as the last option in the ``options'' field of the Reconfigure-init message.

Typically, the server will not provide more than an ORO and / or Authentication option, since it will provide the new configuration information as part of the Request/Reply transaction triggered by the Reconfigure-init message.

The server may either unicast the Reconfigure-init message to one client or multicast the message to one or more Reconfigure Multicast Addresses previously sent as options to the clients. The server may unicast Reconfigure-init messages to more than one client concurrently; for example, to reliably reconfigure all clients, the server will unicast a Reconfigure-init message to each client.

If the server unicasts to one or more clients, it waits for a Request message from those clients confirming that it has received the Reconfigure-init and are thus initiating a Request/Reply transaction with the server. The server can determine that a Request message is in response to a Reconfigure-init because the transaction-ID in the Request will be the same value as was used in the Reconfigure-init message.

If the server multicasts the Reconfigure-init message, it must use some TBD authentication mechanism that can authenticate the server to multiple clients. There is no reliability mechanism for multicast Reconfigure-init messages. A server might use multicast in the case where it does not have a list of its clients; for example, a server that distributes configuration information to clients using stateless autoconfiguration might not keep a list of clients it has communicated with.

12.4.5. Time out and retransmission of Reconfigure-init messages

It the server does not receive a Request message from the client in RECREP_MSG_TIMEOUT milliseconds, the server retransmits the Reconfigure-init message, doubles the RECREP_MSG_TIMEOUT value and waits again. The server continues this process until REC_MSG_ATTEMPTS unsuccessful attempts have been made, at which point the server SHOULD abort the reconfigure process.

Default and initial values for RECREP_MSG_TIMEOUT and REC_MSG_ATTEMPTS are documented in section 3.5.

12.4.6. Receipt of Request messages

The server generates and sends Reply message(s) to the client as described in section 11.6.3, including in the ``option'' field new values for configuration parameters.

12.5. Client Behavior

A client MUST always monitor UDP port 546 for Reconfigure-init messages on interfaces upon which it has acquired DHCP parameters. Since the results of a reconfiguration event may affect application layer programs, the client SHOULD log these events, and MAY notify these programs of the change through an implementation-specific interface.

12.5.1. Receipt of Reconfigure-init messages

Upon receipt of a valid Reconfigure-init message, the client initiates a Request/Reply transaction with the server.

12.5.2. Creation and sending of Request messages

When responding to a Reconfigure-init, the client creates and sends the Request message in exactly the same manner as outlined in section 11.4.1 with the following differences:

transaction-ID
: The client copies the transaction-ID from the Reconfigure-init message into the Request message.

IAs
: The client includes IA options containing the addresses the client currently has assigned to those IAs for the interface through which the Reconfigure-init message was received.

Pause before sending Request
: The client pauses before sending the Request for a random value within the range REC_REP_MIN and REC_REP_MAX seconds. This delay helps reduce the load on the server generated by processing large numbers of triggered Request messages from a multicast Reconfigure-init message.

12.5.3. Time out and retransmission of Request messages

The client uses the same variables and retransmission algorithm as it does with Request messages generated as part of a client-initiated configuration exchange. See section 11.4.2 for details.

12.5.4. Receipt of Reply messages

Upon the receipt of a valid Reply message, the client extracts the contents of the ``option'' field, and sets (or resets) configuration parameters appropriately. The client records and updates the lifetimes for any addresses specified in IAs in the Reply message. If the configuration parameters changed were requested by the application layer, the client notifies the application layer of the changes using an implementation-specific interface.

13. Using DHCP for network renumbering

This section has been deleted (to be moved to ``Notes about DHCP'' doc?).

14. DHCP Client Implementor Notes

This section provides helpful information for the client implementor regarding their implementations. The text described here is not part of the protocol, but rather a discussion of implementation features we feel the implementor should consider during implementation.

14.1. Primary Interface

Since configuration parameters acquired through DHCP can be interface-specific or more general, the client implementor SHOULD provide a mechanism by which the client implementation can be configured to specify which interface is the primary interface. The client SHOULD always query the DHCP data associated with the primary interface for non-interface specific configuration parameters. An implementation MAY implement a list of interfaces which would be scanned in order to satisfy the general request. In either case, the first interface scanned is considered the primary interface.

By allowing the specification of a primary interface, the client implementor identifies which interface is authoritative for non-interface specific parameters, which prevents configuration information ambiguity within the client implementation.

14.2. Advertise Message and Configuration Parameter Caching

If the hardware the client is running on permits it, the implementor SHOULD provide a cache for Advertise messages and a cache of configuration parameters received through DHCP. Providing these caches prevents unnecessary DHCP traffic and the subsequent load this generates on the servers. The implementor SHOULD provide a configuration knob for setting the amount of time the cache(s) are valid.

14.3. Time out and retransmission variables

Note that the client time out and retransmission variables outlined in section 3.5 can be configured on the server and sent to the client through the use of the ``DHCP Retransmission Parameter Option'', which is documented in section 22.6. A client implementation SHOULD be able to reset these variables using the values from this option.

14.4. Server Preference

A client MUST wait for SRVR_PREF_WAIT seconds after sending a DHCP Solicit message to collect Advertise messages and compare their preferences (see section 15.3), unless it receives an Advertise message with a preference of 255. If the client receives an Advertise message with a preference of 255, then the client MAY act immediately on that Advertise without waiting for any more additional Advertise messages.

15. DHCP Server Implementor Notes

This section provides helpful information for the server implementor.

15.1. Client Bindings

A server implementation MUST use the IA's UUID and the prefix specification from which the client sent its Request message(s) as an index for finding configuration parameters assigned to the client. While it isn't critical to keep track of the other parameters assigned to a client, the server MUST keep track of the addresses it has assigned to an IA.

The server should periodically scan its bindings for addresses whose leases have expired. When the server finds expired addresses, it MUST delete the assignment of those addresses, thereby making these addresses available to other clients.

The client bindings MUST be stored in non-volatile storage.

The server implementation should provide policy knobs to control whether or not the lifetimes on assigned addresses are renewable, and by how long.

15.2. Reconfigure-init Considerations

A server implementation MUST provide an interface to the administrator for initiating reconfigure-init events.

A server implementation may provide a mechanism for allowing the specification of how many clients comprise a reconfigure multicast group. This enables the administrator to control the hit a server takes when a reconfigure-init event occurs.

15.3. Server Preference

The server implementation SHOULD allow the setting of a server preference value by the administrator. The server preference variable is an unsigned single octet value (0--255), with the lowest preference being 0 and the highest 255. Clients will choose higher preference servers over those with lower preference values. If you don't choose to implement this feature in your server, you MUST set the server preference field to 0 in the Advertise messages generated by your server.

15.4. Request Message Transaction-ID Cache

In order to improve performance, a server implementation MAY include an in memory transaction-ID cache. This cache is indexed by client binding and transaction-ID, and enables the server to quickly determine whether a Request is a retransmission or a new Request without the cost of a database lookup. If an implementor chooses to implement this cache, then they SHOULD provide a configuration knob to tune the lifetime of the cache entries.

16. DHCP Relay Implementor Notes

A relay implementation SHOULD allow the specification of a list of destination addresses for forwarded messages. This list MAY contain any mixture of unicast addresses and multicast addresses.

If a relay receives an ICMP message in response to a DHCP message it has forwarded, it SHOULD log this event.

17. Open Issues for Working Group Discussion

This section contains some items for discussion by the working group.

17.1. Authentication

Authentication is not discussed in this document.

17.2. DHCP-DNS interaction

Interaction among DHCP servers, clients and DNS servers is not discussed in this document.

17.3. **Release vs.** Decline

Should there be a separate Decline message through which the client informs the server that it has discovered an address that is in use by some other host?

17.4. Request messages

In DHCPv4, there has been much confusion about overloading DHCPREQUEST with the actions of initial address allocation (INIT), address confirmation (INIT-REBOOT), and extending leases (RENEW/REBIND).

The model for DHCPv6 messages described in section 11 also uses one type of message, Request, in each of the scenarios in sections 11.4.8 through 11.4.10. The DHCPv6 specification in this document does not differentiate the actions taken by a server based on different times at which a client might initiate a Request/Reply exchange with a server. That is, the description of server actions in section 11.6.1 does not differentiate among Requests received from clients based on the client behavior described in sections 11.4.8 through 11.4.10.

It may be necessary to define different server behaviors for each of the client scenarios. For example, in the address-reconfirmation scenario (section 11.4.9), servers cannot safely assign new addresses to a client. The reconfirmation Request is broadcast to multiple servers, which cannot coordinate the assignment of any addresses. Therefore, in this scenario, servers can only acknowledge or deny the validity of addresses but cannot allocate any new addresses.

17.5. Use of term ``agent''

The term ``agent'', taken to mean ``relay agent or server'', may be confusing. ``relay agent or server'' might be clearer.

17.6. Use of terms ``subnet'' and ``network''

The term ``subnet'' has been eliminated from the document. The term ``network'' is no longer used to describe a link, collection of links or collection of IPv6 addresses.

18. Security

This document references an ``authentication option'' which is TBD.

DISCUSSION:

Based on the discussion of security issues at the 8/31/00 design team teleconference and subsequent DHC WG mailing list discussion, DHCPv6 will use the security model from DHCPv4, as described in draft-ietf-dhc-authentication-15.txt.

19. Year 2000 considerations

Since all times are relative to the current time of the transaction, there is no problem within the DHCPv6 protocol related to any hardcoded dates or two-digit representation of the current year.

20. IANA Considerations

This document defines message types 1--8 to be received by UDP at port numbers 546 and 547. Additional message types may be defined in the future.

Section 3.1 lists several multicast addresses used by DHCP.

This document also defines several status codes that are to be returned with the Reply and Reconfigure-reply messages (see sections 9.4 and 9.7). The non-zero values for these status codes which are currently specified are shown in the table in section 3.4.

There is a DHCPv6 option described in section 22.6, which allows clients and servers to exchange values for some of the timing and retransmission parameters defined in section 3.5. Adding new parameters in the future would require extending the values by which the parameters are indicated in the DHCP option. Since there needs to be a list kept, the default values for each parameter should also be stored as part of the list.

All of these protocol elements may be specified to assume new values at some point in the future. New values should be approved by the process of IETF Consensus [10].

21. Acknowledgments

Thanks to the DHC Working Group for their time and input into the specification. Ralph Droms and Thomas Narten have had a major role in shaping the continued improvement of the protocol by their careful reviews. Many thanks to Matt Crawford, Erik Nordmark, Gerald Maguire, and Mike Carney for their studied review as part of the

Last Call process. Thanks also for the consistent input, ideas, and review by (in alphabetical order) Brian Carpenter, Jack McCann, Yakov Rekhter, Matt Thomas, Sue Thomson, and Phil Wells.

Thanks to Steve Deering and Bob Hinden, who have consistently taken the time to discuss the more complex parts of the IPv6 specifications.

22. DHCP options

Options are used to carry additional information and parameters in DHCP messages. Every option shares a common base format, as described in section 22.1.

this document describes the DHCP options defined as part of the base DHCP specification. Other options may be defined in the future in a separate document.

22.1. Format of DHCP options

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          option-code          |           option-len          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          option-data                          |
|                      (option-len octets)                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

option-code
: An unsigned integer identifying the specific option type carried in this option.

option-len
: An unsigned integer giving the length of the data in this option in bytes.

option-data
: The data for the option; the format of this data depends on the definition of the option.

22.2. Identity association option

The identity association option is used to carry an identity association, the parameters associated with the IA and the addresses assigned to the IA.

The format of the IA option is:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              TBD              |            variable           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            IA UUID                            |
|                          (8 octets)                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                              T1                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                              T2                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   num-addrs   |              IPv6 address                     |
+-+-+-+-+-+-+-+-+              (16 octets)                      |
|                                                               |
|                                                               |
+               +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               |   pref. len   |      preferred lifetime       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| pref. lifetime (cont.)        |        valid lifetime         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| valid lifetime (cont.)        |         IPv6 address          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+                               |
|                              ...                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

option-code
    TBD

option-len
    Variable; equal to 17 + num-addrs*25

IA UUID
    The unique identifier for this IA; chosen by the client

T1
    The time at which the client contacts the server from which the addresses in the IA were obtained to extend the lifetimes of the addresses assigned to the IA.

T2 The time at which the client contacts any available server to extend the lifetimes of the addresses assigned to the IA.

num-addrs
An unsigned integer giving the number of addresses carried in this IA option (MAY be zero).

IPv6 address
An IPv6 address assigned to this IA.

preferred lifetime
The preferred lifetime for the associated IPv6 address.

valid lifetime
The valid lifetime for the associated IPv6 address.

The ``IPv6 address'', ``preferred lifetime'' and ``valid lifetime'' fields are repeated for each address in the IA option (as determined by the ``num-addrs'' field).

DISCUSSION:

The details of the format and the selection of an IA's UUID are TBD.

DISCUSSION:

An IA has no explicit ``lifetime'' or ``lease length'' of its own. When the lifetimes of all of the addresses in an IA have expired, the IA can be considered as having expired. T1 and T2 are included to give servers explicit control over when a client recontacts the server about a specific IA.

22.3. Option request option

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          option-code          |           option-len          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    requested-option-code-1    |    requested-option-code-2    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                              ...                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

option-code TBD.

option-len
Variable; equal to twice the number of option codes carried in this option.

option-data
A list of the option codes for the options requested in this option.

22.4. Client message option

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          option-code          |           option-len          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      DHCP client message                      |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

option-code TBD

option-len
Variable; equal to the length of the forwarded DHCP client message.

option-data
The message received from the client; forwarded verbatim to the server.

22.5. Server message option

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          option-code          |           option-len          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      DHCP server message                      |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

option-code TBD

option-len
: Variable; equal to the length of the forwarded DHCP server message.

option-data
: The message received from the server; forwarded verbatim to the client.

22.6. Retransmission parameter option

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          option-code          |           option-len          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          option-data                          |
|                      (option-len octets)                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

option-code
: An unsigned integer identifying the specific option type carried in this option.

option-len
: An unsigned integer giving the length of the data in this option in bytes.

option-data
: The data for the option; the format of this data depends on the definition of the option.

22.7. Authentication option

The authentication option is TBD.

23. Changes in this draft

This section describes the changes between this version of the DHCPv6 specification and draft-ietf-dhc-dhcpv6-15.txt.

23.1. Order of sections

New sections have been added at the end of this document to minimize changes in section numbering. Those sections will be rearranged in a future revision.

23.2. Reconfigure message

DHCP Reconfigure and Reconfigure-reply messages and the associated mechanisms have been removed from this draft of the specification.

23.3. Releasable resources

``Releasable resources'' have been removed from this draft.

23.4. DHCP message header

A common fixed DHCP message header has been defined. Not all fields are used in all messages.

23.5. Design goals

The second sentence in the 8th design goal bullet has been removed.

23.6. Overview

Section 8.2 (DHCP agents) has been removed. DHCP clients no longer need to know about specific DHCP agents.

Section 8.3 has been modified to reflect the new encapsulating mechanism through which relays forward client messages to servers.

Section 8.6 and 8.7 have been modified to describe ``identity associations''.

Section 8.8 has been modified to reflect the deletion of ``reconfigure'' and ``reconfigure-reply'' messages.

23.7. Message formats, 9

Message formats have been changed. All messages share a common fixed message header followed by options. The various control bits (``P'', ``C'') have been removed from the message header.

23.8. Solicit and Advertise messages, (section 10)

The description of the message exchanges have been changed to reflect:

- New relay behavior - encapsulated client messages

- Use of IAs

23.9. Prefix advertisement

Servers no longer advertise prefixes.

23.10. Identity Associations

Section 9.11 describes IAs in detail. A definition of ``IA'' has been added to section 2. The description of messages exchanges have been extended to include IAs. The IA option is defined in section 22.2

23.11. Extensions renamed options; defined in this document

``extensions'' are now called ``options''; the options referenced in this document are defined in section 22.

23.12. Transaction-ID ranges

Solicit, Advertise, Request, Reply, Release and Reconfigure-init messages all use an unsigned 16-bit integer ``Transaction-ID''. Transaction-IDs generated by clients are considered to be chosen from a different namespace than those chosen by servers. There is no need to restrict clients and servers to select Transaction-IDs from specific ranges to avoid conflicts.

23.13. Release messages and relays

Release/Reply messages are forwarded through relays. This mechanism eliminates the need for an 'R' bit.

23.14. Discovering relay agents

Clients no longer learn the identity of relay agents. When the client only has a link-local address (e.g., the client has no

assigned addresses), it now multicasts Request message, which is then forwarded by a relay agent on the same link.

A. Comparison between DHCPv4 and DHCPv6

This appendix is provided for readers who will find it useful to see a model and architecture comparison between DHCPv4 [6, 1] and DHCPv6. There are three key reasons for the differences:

- o IPv6 inherently supports a new model and architecture for communications and autoconfiguration of addresses.
- o DHCPv6 benefits from the new IPv6 features.
- o New features were added to support the expected evolution and the existence of more complicated Internet network service requirements.

IPv6 Architecture/Model Changes:

- o The link-local address permits a node to have an address immediately when the node boots, which means all clients have a source IP address at all times to locate an on-link server or relay.
- o The need for BOOTP compatibility and the broadcast flag have been removed.
- o Multicast and address scoping in IPv6 permit the design of discovery packets that would inherently define their range by the multicast address for the function required.
- o Stateful autoconfiguration has to coexist and integrate with stateless autoconfiguration supporting Duplicate Address Detection and the two IPv6 lifetimes, to facilitate the dynamic renumbering of addresses and the management of those addresses.
- o Multiple addresses per interface are inherently supported in IPv6.
- o Some DHCPv4 options are unnecessary now because the configuration parameters are either obtained through IPv6 Neighbor Discovery or the Service Location protocol [15].

DHCPv6 Architecture/Model Changes:

- o The message type is the first byte in the packet.

- o IPv6 Address allocations are now handled in a message option as opposed to the message header.

- o Client/Server bindings are now mandatory and take advantage of the client's link-local address to always permit communications either directly from an on-link server, or from a off-link server through an on-link relay.

- o Servers are discovered by a client Solicit, followed by a server Advertise message

- o The client will know if the server is on-link or off-link.

- o The on-link relay may locate off-link server addresses from system configuration or by the use of a site-wide multicast packet.

- o ACKs and NAKs are not used.

- o The server assumes the client receives its responses unless it receives a retransmission of the same client request. This permits recovery in the case where the network has faulted.

- o Clients can issue multiple, unrelated Request messages to the same or different servers.

- o The function of DHCPINFORM is inherent in the new packet design; a client can request configuration parameters other than IPv6 addresses in the optional option headers.

- o Clients MUST listen to their UDP port for the new Reconfigure message from servers.

- o New options have been defined.

With the changes just enumerated, we can support new user features, including

- o Configuration of Dynamic Updates to DNS

- o Address deprecation, for dynamic renumbering.

- o Relays can be preconfigured with server addresses, or use of multicast.

- o Authentication

- o Clients can ask for multiple IP addresses.

- o Addresses can be reclaimed using the Reconfigure-init message.

- o Integration between stateless and stateful address autoconfiguration.

- o Enabling relays to locate off-link servers.

B. Full Copyright Statement

References

[1] S. Alexander and R. Droms. DHCP Options and BOOTP Vendor Extensions. Request for Comments (Draft Standard) 2132, Internet Engineering Task Force, March 1997.

[2] S. Bradner. Key words for use in RFCs to Indicate Requirement Levels. Request for Comments (Best Current Practice) 2119, Internet Engineering Task Force, March 1997.

[3] S. Bradner and A. Mankin. The Recommendation for the IP Next Generation Protocol. Request for Comments (Proposed Standard) 1752, Internet Engineering Task Force, January 1995.

[4] W. J. Croft and J. Gilmore. Bootstrap Protocol. Request for Comments 951, Internet Engineering Task Force, September 1985.

[5] S. Deering and R. Hinden. Internet Protocol, Version 6 (IPv6) Specification. Request for Comments (Draft Standard) 2460, Internet Engineering Task Force, December 1998.

[6] R. Droms. Dynamic Host Configuration Protocol. Request for Comments (Draft Standard) 2131, Internet Engineering Task Force, March 1997.

[7] R. Hinden and S. Deering. IP Version 6 Addressing Architecture. Request for Comments (Proposed Standard) 2373, Internet Engineering Task Force, July 1998.

[8] S. Kent and R. Atkinson. IP Authentication Header. Request for Comments (Proposed Standard) 2402, Internet Engineering Task Force, November 1998.

[9] J. McCann, S. Deering, and J. Mogul. Path MTU Discovery for IP version 6. Request for Comments (Proposed Standard) 1981, Internet Engineering Task Force, August 1996.

[10] T. Narten and H. Alvestrand. Guidelines for Writing an IANA Considerations Section in RFCs. Request for Comments (Best Current Practice) 2434, Internet Engineering Task Force, October 1998.

[11] T. Narten, E. Nordmark, and W. Simpson. Neighbor Discovery for IP Version 6 (IPv6). Request for Comments (Draft Standard) 2461, Internet Engineering Task Force, December 1998.

[12] D. C. Plummer. Ethernet Address Resolution Protocol: Or converting network protocol addresses to 48.bit Ethernet address for transmission on Ethernet hardware. Request for Comments (Standard) 826, Internet Engineering Task Force, November 1982.

[13] J. Postel. User Datagram Protocol. Request for Comments (Standard) 768, Internet Engineering Task Force, August 1980.

[14] S. Thomson and T. Narten. IPv6 Stateless Address Autoconfiguration. Request for Comments (Draft Standard) 2462, Internet Engineering Task Force, December 1998.

[15] J. Veizades, E. Guttman, C. Perkins, and S. Kaplan. Service Location Protocol. Request for Comments (Proposed Standard) 2165, Internet Engineering Task Force, June 1997.

[16] P. Vixie, Ed., S. Thomson, Y. Rekhter, and J. Bound. Dynamic Updates in the Domain Name System (DNS UPDATE). Request for Comments (Proposed Standard) 2136, Internet Engineering Task Force, April 1997.

Chair's Address

The working group can be contacted via the current chair:

Ralph Droms
Cisco Systems
300 Apollo Drive
Chelmsford, MA 01824

Phone: (978) 244-4733
E-mail: rdroms@cisco.com

Author's Address

Questions about this memo can be directed to:

Jim Bound
Compaq Computer Corporation
Mail Stop: ZK03-3/U14
110 Spitbrook Road
Nashua, NH 03062
USA
Phone: +1-603-884-0400
Email: bound@zk3.dec.com

Mike Carney
Sun Microsystems, Inc
Mail Stop: UMPK17-202
901 San Antonio Road
Palo Alto, CA 94303-4900
USA
Phone: +1-650-786-4171
Email: mwc@eng.sun.com

Charles E. Perkins
Communications Systems Lab
Nokia Research Center
313 Fairchild Drive
Mountain View, California 94043
USA
Phone: +1-650 625-2986
EMail: charliep@iprg.nokia.com
Fax: +1 650 625-2502